Claims

1. A method of configuring a product comprising a number of components, the method comprising:

- providing, for each component, information relating to a group of alternatives for the component,
- defining rules relating to compatibilities between alternatives from different components,
- representing the rules in a Directed Acyclic Graph (DAG), and
- iteratively configuring the product by repeatedly:
  - choosing a component,
  - selecting an alternative from this component's group of alternatives,
  - checking the DAG whether the alternative selected is compatible with other chosen alternatives from other components.

2. A method of configuring a product according to claim **1** in which the iterative configuring is ended when an alternative is chosen for each component and when the chosen alternatives of the components are compatible.

3. A method of configuring a product according to claim **1**, wherein the step of selecting the alternative, and before the selection of the alternative, comprises:

- using the DAG to determine, for at least one of the components, a subset of alternatives for the component, so that each of the alternatives in the subset is compatible with the chosen alternatives from the other components, and
- providing this information to a user.

4. A method of configuring a product according to claim claim **3**, wherein the method further comprises providing a system with a speech synthesizer and the providing of information to a user further comprises

- providing the information by speech generated by the speech synthesizer.

5. A method of configuring a product according to claim **1**, wherein the steps of choosing a component and the alternative further comprise, for each of the components:

- using the DAG to check which of the alternatives of the component that are compatible with at least one of the chosen alternatives of each of the other components,
- providing a user with this information,

- allowing the user to select one of the alternatives that were compatible with at least one of each of the other component's chosen alternatives.

6. A method according to claim **1**, wherein the steps of selecting the alternative and checking the DAG further comprise the steps of:

- selecting or defining a subgroup of alternatives to the chosen component,
- checking the DAG for which of the alternatives in the subgroup that are compatible with chosen alternatives from other components, and
- providing information relating to which of the alternatives in the subgroup are compatible with chosen alternatives of other components.

7. A method of configuring a product according to claim **1**, wherein the iterative configuration further comprises:

- at least once, defining information relating to limiting the alternatives of at least one of the components, and
- checking the DAG for which of the alternatives of the components is compatible with the limiting information.

8. A method of configuring a product according to claim **1** in which the iterative configuring is ended upon request from a user, and information is provided relating to all possible compatible products comprising at least one chosen alternative for each of the products for which an alternative is chosen, and this information is provided to the user.

9. A method of configuring a product according to claim **1** in which the iterative configuring comprises the steps of obtaining a number of all possible compatible products comprising at least one chosen alternative for each of the products for which an alternative is chosen, and providing this information to the user.

10. A method according to claim **1**, wherein the step of representing the rules in the DAG comprises representing the rules in a graph comprising:

- at least one terminal node,
- a plurality of nodes comprising:
  - a mathematical expression having a plurality of possible disjoint outcomes and
  - a number of pointers corresponding to the number of possible outcomes of the expression,

wherein:

- a pointer of at least one of the nodes points to another of the nodes,
- a pointer of at least one of the nodes points to one of the at least one terminal node, and
- at least one of the nodes being a top-most node from which one or more paths are defined from a top-most node to one of the at least one terminal node via one or more of the nodes and the pointers thereof, each node being part of at least one path.

11. A method of configuring a product according to claim **10**, wherein the step of representing the rules in the DAG comprises providing one or more of the nodes with mathematical expressions each comprising a mathematical operator, each operator describing how the rules represented by the nodes pointed to by the pointers of the pertaining node are to be combined in order to represent the combined set of rules.

12. A method of configuring a product according to claim **10**, wherein the step of representing the rules in the DAG comprises representing the rules in a DAG comprising a number of nodes, the mathematical expression of which is a Boolean expression.

13. A method of configuring a product according to claim **10**, wherein the step of representing the rules in the DAG comprises representing the rules in a DAG comprising a number of nodes each comprising a mathematical expression which is a variable.

14. A method of configuring a product according to claim **10**, wherein the step of representing the rules in the DAG comprises representing the rules in a DAG comprising nodes, the mathematical expressions of which are ordered according to a given ordering such that, for each node, the expression of the actual node is of a lower order than the expressions of any nodes pointed to by the pointers of the actual node.

15. A method of configuring a product according to claim **10** wherein the representing of the rules in the DAG further comprises the steps of:

- identifying a first and a second node having the same expression and the pointers of which point to the same nodes, and
- having pointers pointing to the first node point to the second node.

16. A method of configuring a product according to claim **10**, wherein the step of representing the rules the DAG comprises:

- representing each rule as a logical expression,
- from each logical formula constructing a partial DAG representing the set of possible solutions to the formula,
- constructing the DAG representing all the rules from the partial DAGs representing each of the logical formulas.

17. A method of configuring a product according to claim **16**, wherein the step of providing the information relating to the alternatives for each component comprises:

- selecting Boolean variables for representing the individual alternatives of the component,
- providing an encoding for each of the alternatives of the component as a combination of Boolean values for the Boolean variables.

18. A method according to claim **17**, wherein the step of representing each rule as a logical formula/expression comprises providing the Boolean variables relating to the alternatives to which the rule relates and interrelating the variables in accordance with the rule.

19. A method according to claim **10**, wherein the step of representing the rules in the DAG comprises providing at least one type of terminal node and wherein, for each path comprising a such terminal node, the combination of all expressions and all pertaining outcomes relating to the pointers of the path relate to either compatible products or non-compatible products.

20. A method of configuring a product according to claim **19**, wherein the step of representing the rules in the DAG comprises providing a first and a second type of terminal nodes, and wherein:

- for each path comprising a terminal node of the first type, the combination of all expressions and all pertaining outcomes relating to the pointers of the path relate to a compatible product, and
- for each path comprising a terminal node of the second type, the combination of all expressions and all pertaining outcomes relating to the pointers of the path relate to a non-compatible product.

21. A method according to claim **20**, wherein the first type of terminal node is adapted to represent "true", "one" or "1", and wherein the second type of terminal node is adapted to represent "false", "zero" or "0".

22. A method according to claim **20**, wherein the step of representing the rules in the DAG comprises:

- representing each rule as a logical expression,
- from each logical formula constructing a partial DAG representing the set of possible solutions to the formula,
- constructing the DAG representing all the rules from the partial DAGs representing each of the logical formulas,

the step of providing the information relating to the alternatives for each component comprises:

- selecting Boolean variables for representing the individual alternatives of the component,
- providing an encoding for each of the alternatives of the component as a combination of Boolean variables,

and the step of selecting an altemative comprises:

- identifying Boolean variables relating to the other alternative(s) of the component and nodes comprising expressions relating to such other alternative(s), and
- in the DAG, identifying paths comprising such nodes and altering any terminal node(s) thereof of the first type to terminal node(s) of the second type.

23. A method of configuring a product according to claim **20** in which the step of iteratively configuring the product comprises the steps of:

- obtaining a number of all possible compatible products comprising at least one chosen alternative for each of the products for which an alternative is chosen,
- providing this information to the user,

and wherein the computing of the number of possibilities of different choices is performed by the following steps applied to the DAG and for each top-most node:

- starting from the topmost node and iteratively finding the number of possibilities represented by the actual node, by performing the steps of:
  - if the node is a terminal node, providing a "1" if the terminal node is of the first type and a "0" if it is of the second type,
  - else: finding the number of possibilities represented by each node pointed to by a pointer of the actual node, and therefrom computing the number of possibilities represented by the node.

24. A method of configuring a product according to claim **1**, wherein, the step of checking the DAG further comprises, if the selected alternative is not compatible with other chosen alternatives,

- providing information relating to other chosen alternatives which are not compatible with the selected alternative, and
- providing this information to a user.

25. A method of configuring a product according to claim **1**, wherein the step of defining the rules comprises:

- obtaining, by querying a database, information relating to alternatives relating of one or more components and/or information relating to compatibility between two or more altematives to different components, and
- building one or more rules from the information obtained from the database.

26. A method of configuring a product according to claim **25**, wherein the database comprises a two-dimensional table having, in each of a plurality of rows thereof, information relating to a product comprising an alternative from each component, the alternatives being compatible, wherein the step of providing a rule comprises providing a rule relating to the information of each row, and wherein the step of representing the rules in the DAG comprises providing a disjunction of the rules.

27. A method of configuring a product according to claim **10**, wherein the step of checking the DAG whether an alternative is compatible comprises searching the DAG for a path from a topmost node to a terminal node, the search comprising:

- starting with the top-most node as an actual node,
- iteratively, until the actual node is a terminal node:
  - evaluating the mathematical expression in the actual node and determining the outcome thereof in view of the alternatives chosen from other components,
  - selecting the pointer of the node representing the outcome,
  - selecting, as the actual node, the node pointed to by the selected pointer,
- providing information relating to the chosen alternatives, and
- the information relating to the path represents that the choices are compatible.

28. A method of configuring a product according to claim **20**, wherein information is provided from a path in the DAG by providing, from the expressions of the nodes of the path, information relating to which alternative(s) of a given component has/have been chosen, and the information of compatibility of the product comprising those alternatives is given by the representation of the terminal node of the path.

29. A method of configuring a product according to claim **28**, wherein the expressions of nodes of the DAG are Boolean variables, the terminal nodes represent either "true"or "false", the information of a path relating to the identities of the variables in the mathematical expression(s) of the node(s) of the path and values or dependencies thereof, the identities and values/dependencies relating to chosen alternatives of components, the chosen components being compatible if the terminal node of the path represents "true" and the chosen components being incompatible if the terminal node of the path represents "false".

30. A method of configuring a product according to claim **10**, wherein the step of representing the rules in the DAG comprises:

- representing the rules in an actual DAG,
- selecting at least one of the components to be hidden,
- changing the actual DAG by:

- identifying nodes in the actual DAG comprising expressions relating to the selected component(s),
- removing these nodes from the actual DAG,
- adding nodes, not comprising expressions relating to the selected component(s), to the actual DAG so that the compatibilities implied by these component(s) are reflected by the actual DAG,

- providing the actual DAG as the DAG representing the rules.

31. A method of configuring a product according to claim **10** wherein the step of representing the rules in the DAG comprises:

- for each of the rules, constructing a partial DAG representing the rule,
- identifying at least one of the components to be hidden,
- selecting an ordering of the identified components,
- initially constructing an actual DAG representing no rules, and then repeatedly,
  - selecting a non-selected component of lowest order,
  - repeatedly, until all partial DAGs comprising expressions relating to the selected component have been chosen:
    * choosing a partial DAG comprising expressions relating to the selected component,
    * combining the actual DAG with the chosen partial DAG into a new actual DAG,
  - changing the actual DAG by:
    * identifying nodes in the actual DAG comprising expressions relating to the identified component,
    * removing these nodes from the actual DAG,
    * adding nodes, not comprising expressions relating to the identified component, to the actual DAG so that the compatibilities implied by the identified component are reflected by the actual DAG,
- providing the DAG by combining the actual DAG with all non-chosen partial DAGs.

32. A method of configuring a product according to claim **1**, the method further comprising:

- identifying a user,
- performing the step of selecting an alternative of a component by the user through communication between a device controlled by the user and another device where the iterative configuration is performed,

- transmitting information relating to the checking of the DAG to the user.

33. A method of configuring a product according to claim **1**, wherein the method further comprises:

- identifying a user,
- prior to the iterative configuring:
  - transmitting the DAG to a device controlled by the user,
  - performing the iterative configuring on the user's device.

34. A method of configuring a product according to claim **1**, further comprising the steps of, during the iterative configuration:

- obtaining information relating to one or more alternatives for components for which no alternatives have been chosen, each of the one or more alternatives being compatible with the chosen alternatives, and
- providing the user with this information.

35. A method of configuring a product according to claim **1**, wherein the method further comprises providing a system with a speech recognizer, and wherein the step of iteratively configuring the product further comprises

- choosing a component from a text recognized by the speech recognizer, and
- selecting an alternative from this component's group of alternatives from a text recognized by the speech recognizer.

36. A method of configuring a product according to claim **1**, wherein the method further comprises identifying a configurable device and an interface device, and

- storing the DAG representing the rules on the configurable device,
- uploading the DAG from the configurable device to the interface device, and
- in the step of iteratively configuring the product, performing the checking of the DAG whether the alternative selected is compatible with other chosen alternatives from other components on the interface device.

37. A method of configuring a product according to claim **36** wherein the method further comprises identifying a list of predetermined components in the configurable device and identifying a list of predetermined alternatives for these components in the configurable device, and wherein the step of iteratively configuring the product further comprises

- performing the checking of the DAG whether the alternative selected is compatible with other chosen alternatives from other components and compatible with the predetermined alternatives on the interface device.

38. A method of configuring a product according to claim **1**, wherein the method further comprises identifying a list of observer components and a list of non-observer components, and

- representing the rules for the non-observer components in a DAG,
- determining, for each observer component, a subset of the rules, such that from these rules it is possible to determine the alternatives for the observer component that are compatible with alternatives for the non-observer components,
- representing for each observer component the subset of rules as an observer DAG, and
- in the step of iteratively configuring the product
  - checking the DAG whether the alternative selected is compatible with other chosen alternatives from other components,
  - determining a set of system determined alternatives by determining for each component whether there is only a single alternative compatible with all the chosen alternatives,
  - for at least one of the observer components, checking the observer DAG for the observer component to determine whether there is only a single alternative compatible with other chosen alternatives and the set of system determined alternatives, and
  - providing this information to a user.

39. A method of configuring a product according to claim **1** wherein the step of iteratively configuring the product further comprises

- for each pair of component and alternative providing a classification of the state of the pair,
- adopting the classification to one of a list of outcomes comprising blocked, selectable, user selected, system selected, or forceable,
- providing a classification of blocked when the alternative cannot be chosen for the component even without considering choices of alternatives for other components,
- providing a classification of selectable when the alternative for the component is compatible with the chosen alternatives from the other components,
- providing a classification of user selected when the alternative has already been chosen for the component,

- providing a classification of system selected when the alternative is the only choice for the component that is compatible with the chosen alternatives from the other components and the alternative has not been chosen by the user,
- providing a classification of forceable when the alternative can be chosen for the component but is incompatible with some of the other choices of alternatives of the other components, and
- providing information on the classification to a user.

40. A computer program comprising computer program code means adapted to perform all the steps of the method of claim **1** when said program is run on a computer.

41. A computer program as claimed in claim **40** embodied on a computer-readable medium.

42. A computer readable medium comprising the computer program according to claim **40**.